AN

INTERNSHIP REPORT

ON

STAY SAFE WOMEN SECURITY ANDROID APP PROJECT REPORT

BY

KAMAL ACHARYA

(Tribhuvan University)

Date: 2025/04/10

CHAPTER-1

ABSTRACT

Women’s security is a critical issue in today’s world and it’s very much needed for every individual to be acting over such an issue. This document describes a GPS based “Women Security System" that provides the combination of GPS devices as well as provide alerts and messages with an emergency button trigger whenever somebody is in trouble They might not have so much time, all that they have to do is generate a distress emergency signal by shaking up their phone. Our system provides a realizable, cost effective solution to problem detection. Nowdays due to recently happened cases such as rape by drivers or colleagues, burglary etc., women security, especially women security has become the foremost priority of the world. System uses the Global Positioning System (GPS) technology to find out the location of women. The information of women's position provided by the device can be viewed on Google maps using Internet or specialized software. The companies are looking for-ward to the security problem and require a system that will efficiently evaluate the problem of women security working in night shifts, traveling alone. We focus on the proposed model that can be used to deal with the security issue of women using GPS based tracking systems.

CHAPTER-2

INTRODUCTION

Women are adept at mobilizing diverse groups for a common cause. They often work across ethnic, religious, political, and cultural divides to promote peace. We are all aware about the importance of the safety of women but we must realize that they should be properly protected. Women are not as physically strong as men, in an emergency situation a helping hand would be a relief for them. The best way to minimize your chances of becoming a victim of violent crime (robbery, sexual assault, rape, domestic violence) is to identify and call on resources to help you out of dangerous situations. Whether you're in immediate trouble or get separated from friends during a night out and don't know how to get home, having these apps on your phone can reduce your risk and bring assistance when you need it. Although several were originally developed for students to reduce the risk of sexual assault on campus, they are suitable for all women in the light of recent outrage in Delhi which shook the nation and woke us to the safety issues for our daughters, people are gearing up in different ways to fight back. A host of new apps have been developed to provide security systems to women on their phones.

Here we introduce an app which ensures the safety of women. This helps to identify and call on resources to help the one out of dangerous situations. These reduce risk and bring assistance when we need it and help us to identify the location of the one in danger. This app is designed to provide security to women. The main purpose of this app is to provide awareness on the time of critical situations for women. Generally user can activate this service by adding the emergency contacts using the emergency contacts icon in the app.While in emergency the user would have to shake up his/her handset**,** after that a distress signal(SOS) will automatically got generated from the user end and send SMS to those contacts which has been saved at the time of registration. The SMS contains your message and your exact location.

CHAPTER-3

PROFILE OF THE PROBLEM

3.1 PROJECT PURPOSE

The main purpose of the project is to provide a highly reliable security system for the safety of women. The proposed system is based upon advanced sensors and GPS. The basic aim of the system is to develop a low cost solution for GPS based women
tracking system (Womens' Safety System). The main objective of the system is to track the current location of the person which has an android enabled mobile by extracting the longitude and latitude of that target person.

3.2 RATIONALE OF THE PROJECT

We provide this application where women and other users can use this application to contact the parents and friends in the time of need or in case of any emergency .The application provides a friendly interface to use various other emergency tools at the time of emergency. The application can be used both in online and offline mode. Students and other members having an Android platform can easily use the application. The application provides various tools in the form of buttons so as to provide a friendly interface to the users. The user just needs to tap on the button to use the tools such as a loud alarm button ,texting along with sending the user location and sending the location via the SMS when the end user is not using the Android platform .

3.3 EXISTING SYSTEM

3.3.1 INTRODUCTION

There are certain Women Security Applications which are quite similar to our application.

3.3.2 DRAWBACKS OF THE EXISTING SYSTEM

- Requires good network connectivity.
- Good Android platform.
- Difficult to inform immediately the location of the user in trouble.

3.4 PROPOSED SYSTEM

The proposed system is for women safety and overcomes the disadvantages of the existing systems. This proposed system is a GPS based “Women Security System”. It consists of a GPS device i.e. any Android Phone .The device will provide the position information such as latitude, longitude of the user.

- The proposed system is based on advanced sensors. Whenever the user shakes his/her phone, a distress signal will get generated automatically and then a message alert is sent to the contacts which are added in the emergency contacts list.
- Low battery alert : when the user battery will be less than 10%, a low battery alert message will be sent to the emergency contacts.

3.5 WHAT’S NEW IN THE SYSTEM TO BE DEVELOPED ?

In the new application we are providing a user friendly interface where the user could send the message alert more efficiently and smartly. The user doesn't have to remember all the important contact numbers of siblings, relatives or friends. The new system is also interactive to the users and provides the facility to know their nearby police station , hospitals and their own location.

3.6 GENERAL FUNCTIONALITY

- User-friendly interface.
- Time saving.
- Easy to integrate and access.
- Interactive interface.SMS alerts and notifications will be sent in case of an emergency.

CHAPTER-4

PROBLEM ANALYSIS

4.1 PRODUCT DEFINITION

Women security applications provide a user-friendly interface to their users. This application works in both online and offline mode. Users and other members who have installed this android application can get the help immediately by just shaking up their handset. They can also check the feedback provided by the various users. By clicking on the loud alarm option, it produces a kind of alert sound which makes the other people nearby to that location and they get to know that something wrong happens and they can also help that user. There is also an option of fake caller which helps the user to initiate a fake call into their phone if they want an interruption in the situations where the user feels unsafe.

4.2 FEASIBILITY ANALYSIS

Inputs are required for developing the system, which are stored for the process and for future use. System will work on the inputs given by the user and itself gathers most of the information necessary for its activities. The main objectives that are guiding as in the input stages are:

- Controlling the amount of inputs
- Avoiding inordinate delay
- Controlling errors

Feasibility analysis (FA, also called feasibility study) is used to assess the strengths and weaknesses of a proposed project and present directions of activities which will improve a project and achieve desired results. The nature and components of feasibility studies depend primarily on the areas in which analyzed projects are implemented.

As the name implies, a feasibility study is used to determine the viability of an idea. The objective of such a study is to ensure a project is legally and technically feasible and economically justifiable. It tells us whether a project is worth the investment. It is used to select the best system

that meets the performance requirements. It involves preliminary investigation of the project and examines whether the designed system will be useful to the users. By doing the research beforehand, companies can save money and resources in the long run by avoiding projects that are not feasible.

4.3 TECHNICAL FEASIBILITY

A study of resource availability that may affect the ability to achieve an acceptable system. Technical feasibility is the most difficult area to ensure at the initial stage. Since the objectives, functions, performance cannot be predicted to its fullest, everything seems possible, provided the right assumptions are made.

It is essential that the process of analysis and definition can be conducted in parallel with an assessment of technical feasibility. The consideration that is normally associated with technical feasibility includes resource availability at the organization where the project is to be developed and implemented.

4.4 OPERATION FEASIBILITY

It deals with the consideration about working of the system after installation. The proposed system would be beneficial to its users as their needs are fully satisfied. As this project satisfies all the requirements of the users it is operationally feasible. All the operational aspects are considered carefully here. Only by spending time evaluating feasibility will we be able to reduce the chances for extreme embracement at later stages of a project. The benefits of proposed system are:-

- Ability to handle large amount of a data
- Fast and accurate information is possible
- Security features based on user roles
- Easy Report generation

Thus, considering the above facts management feels that the project is feasible.

4.5 ECONOMICAL FEASIBILITY

The purpose of an Economic Feasibility Study (EFS) is to demonstrate the net benefit of a proposed project for accepting or disbursing electronic funds/benefits, taking into consideration the benefits and costs to the agency, other state agencies, and the general public as a whole i.e Cost Benefit Analysis.

- Resource cost is based on the estimated resources within the technical analysis
- Employee costs should be based on salaries and overhead
- Any hardware or software that you purchase should be listed as well
- Additional costs (if any): This section is an assessment of additional costs incurred from licensing, contracting, out-sources testing, and so on. Cost of maintenance of equipment.

CHAPTER-5

PROJECT PLAN

Project planning defines the project activities and end products that will be performed and describes how the activities will be accomplished. The purpose of the project planning is to define each major task, estimate the time and resources required, and provide a framework for management review and control. The project planning activities and goals include defining :

- The specific work to be performed and goals that define and blind the project.
- Estimates to be documented for planning, tracking, and controlling the project.
- Commitments that are planned, documented, and agreed to by affected groups.
- Project alternatives, assumptions and constraints.

Project Plan Table

Month	Activity
January	Feasibility Study And Analysis
February	Requirement Gathering
March	Implementation
April	Testing And Documentation

Table 5.1

Gantt Chart :

A chart in which a series of horizontal lines shows the amount of work done or production completed in certain periods of time in relation to the amount planned for those periods. The complete Gantt chart of our work flow over the period of four months is as shown below :

Project Plan Gantt Chart

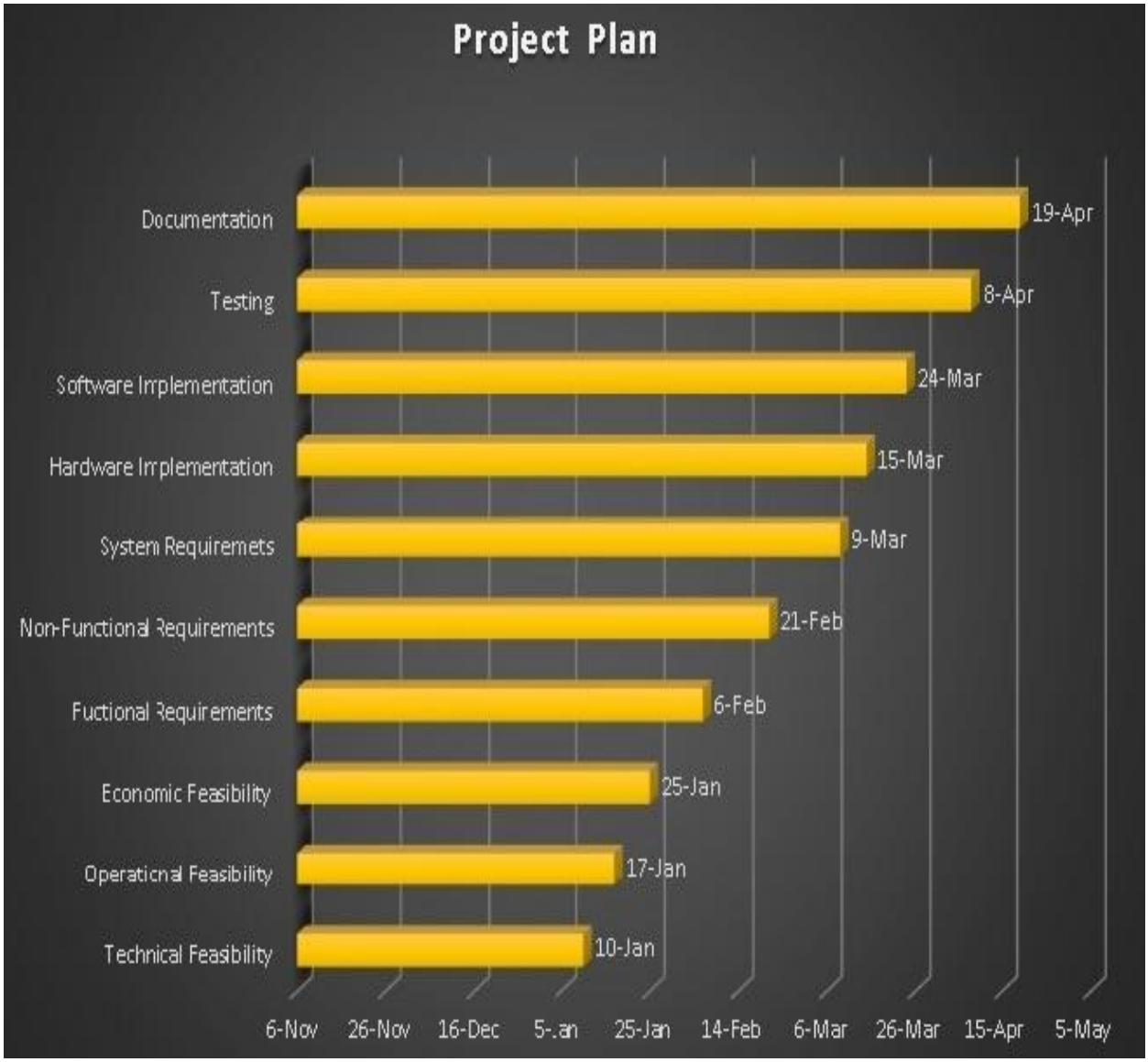


Fig 5.2

CHAPTER-6

SOFTWARE REQUIREMENT ANALYSIS

6.1 PURPOSE

This document describes the software requirements and specification for an Android Application i.e Stay Safe.

6.2 INTENDED AUDIENCE AND READING SUGGESTIONS

The document is intended for all the customers and the developers. The reader is assumed to have basic knowledge of an Android Application. Knowledge and understanding through diagrams is also required.

6.3 PRODUCT FUNCTIONS

1. Scream Alarm : It is perfect for the females as well as other users that need some kind of safety alarm in case they find out that someone is following or stalking them. It also consists of two other types of scream alarm. It's an initial distraction which will buy some time and allow the user to escape from the trouble.

- Male voice scream
- Police siren.

The user could select one of his/her choices from the "Settings" of the application, as keeping in mind the two other scream alarms are also added in this application as nowadays safety and security is everybody's concern.

2. Fake Call Timer : The fake call timer allows the user to make fake calls in the time of need. It helps users to escape from an undesirable situation citing an important call from anyone who needs him/her urgently and rest depends upon user creativity. This feature also helps the user to escape from boring social events

In order to make a fake call the user has to select the “Fake Call” icon and after that user can write any name from which he/she wants a fake call. Users could also set up the timer as per the requirement. The user could also set the default timer from the “Settings” icon of the application. In a critical situation, the user just has to long press the fake call button and automatically get a fake call as per the desired selected timer in the settings.

3. Where Are You : Your friend is out somewhere for a late night party. How could you check where that respective person is ?.Where are your features that allow the user to see the recent location of the friends and family when needed without disturbing the person being tracked.
While the first request is sent by the sender. The sender will have to select the “Where Are You” icon and then a new dialog box of “Pick a Friend” will open up. The sender could select any friend and the request will be sent to the receiver. The receiver will accept that request from their end and a message will be sent to the receiver with the present location of the user.

4. Track Me : The track me feature allows the user to view the exact dynamic location of the victim. First users have to send the Track Me request at the receiver's end. The receiver will accept the request and then his/her name will appear on the friends you are tracking on the bottom of the application. The user could select that friend from there and then it will get automatically redirected to the Google maps from where the user could view the exact location of the victim and also where he/she is heading to.

5. Friends List : This list shows all the contact numbers of family and friends which are added by the user through contacts. This could be done by selecting the contact icon on the bottom right corner of the friends list.

6. Settings : The “Settings” function consists of the following features -:

- **Emergency Services :** It allows the Stay Safe Application to send emergency notifications and SMS with the exact location to the emergency contacts.
- **Low Battery Alert :** The low battery alert feature allows the Stay Safe Application to send low battery alerts and SMS to the emergency contacts.

- **Set Scream Sound :** The user could select any scream sound as per the requirement.
- **Fake Call Timer(On Long press) :** The user could set the fake call default timer as per the requirement.

7. Emergency Distress Signal (SOS) : The distress signal will be generated by the user in case of an emergency. In order to generate the distress signal the user has to shake up his/her phone, then a distress signal will appear at the user end with a default timer of 5 sec. In the end a distress signal will be sent to the emergency contacts added by the user at the time of registration. The application sends SMS and user details as well as the exact location of the user through a push notification at the receiver end, before sending a distress signal the user first has to turn on the emergency services from the settings of the application.

6.4 SPECIFIC FUNCTIONAL REQUIREMENTS

External Interface Requirements

Hardware Requirements

- Processor : Snapdragon, Dual Core.
- Memory Space : 50 Mb
- RAM : 512 MB.
- GPS enabled Android Phone

Software Requirements

- Operating System - Android
- API Level - 14 or higher.

- Disk Usage – 20-50 Mb

6.5 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are often called "Quality Attributes" of a system. Evolution qualities, such as testability, maintainability, extensibility and scalability, which are embodied in the static structure of the software system.

6.5.1 GRAPHICAL USER INTERFACE

- The system shall provide use of icons and toolbars.
- Graphical user Interface has been made interactive so that users can feel good while using the application.
- We have provided the proper image of buttons so that users can understand properly.

6.5.2 ACCESSIBILITY

It should be easily accessible from everywhere where the internet is available. Users will be able to access our application even if they do not have an internet connection or if they were previously logged in.

6.5.3 PERFORMANCE

- The product is based on android and can be run on any android version of.
- The product shall take initial loading time depending on internet connection strength which is needed for the new user to login.
- The performance shall depend upon the hardware and the software components of the client/customer i.e. which smartphone and which android version the client is using.

CHAPTER-7

DESIGNING OF THE PROJECT

7.1 SYSTEM DESIGN

In System design the design functions and operations are described in detail, including screen layouts, business rules, process diagrams and other documentations. The output of this stage will describe the new system as a collection of modules or subsystems. The design stage takes as its initial input the requirements identified in the approved requirements document.

7.1.1 LOGICAL DESIGN

The logical design of our system pertains to an abstract representation of data flows, inputs and outputs of the system. In the context of systems design, modeling can undertake the following forms, including:

- Data Flow Diagrams
- Flow Charts

7.1.2 PHYSICAL DESIGN

The physical design relates to the actual input and output process of the system. This is laid down in terms of how data is input into our system, how it is verified/authenticated, how it is processed, and how it is displayed as output.

DESIGN NOTATIONS

DATA FLOW DIAGRAM - LEVEL 0

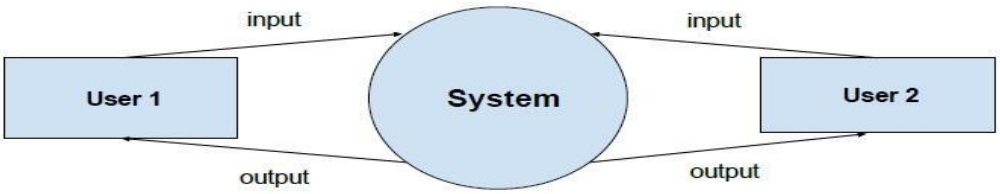


Fig. 7.1

DATA FLOW DIAGRAM - LEVEL 1

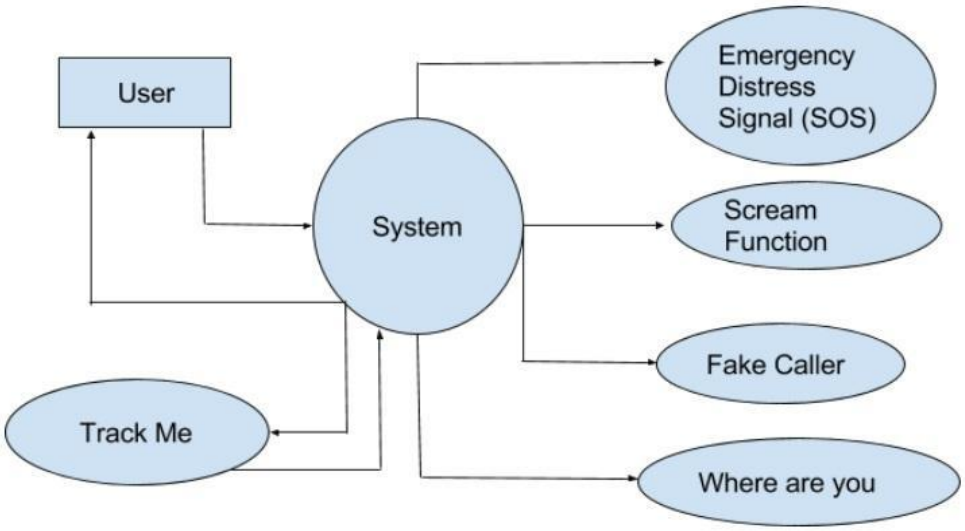


Fig. 7.2

DATA FLOW DIAGRAM - LEVEL 2

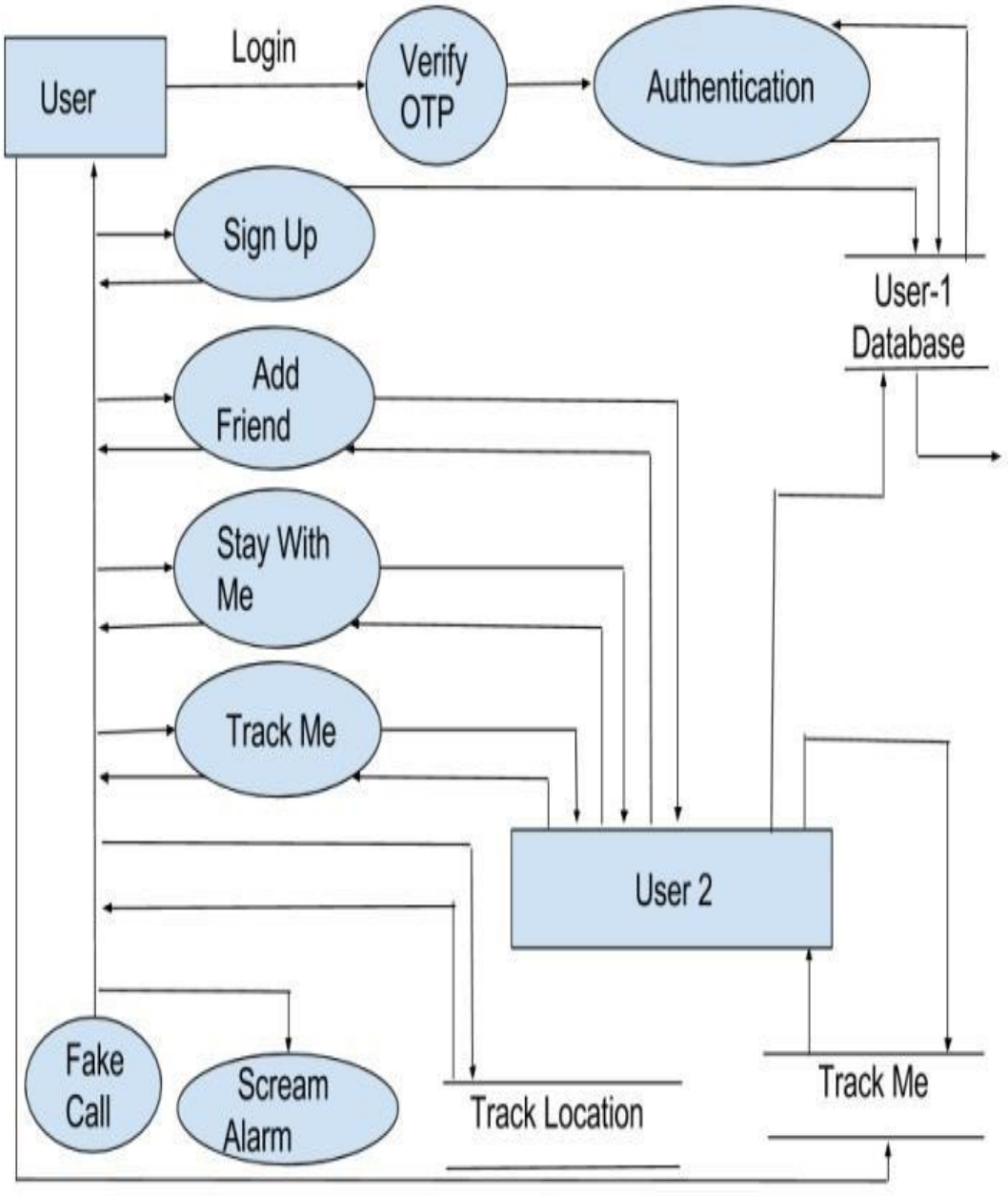


Fig. 7.3

FLOW CHART : FAKE CALL TIMER

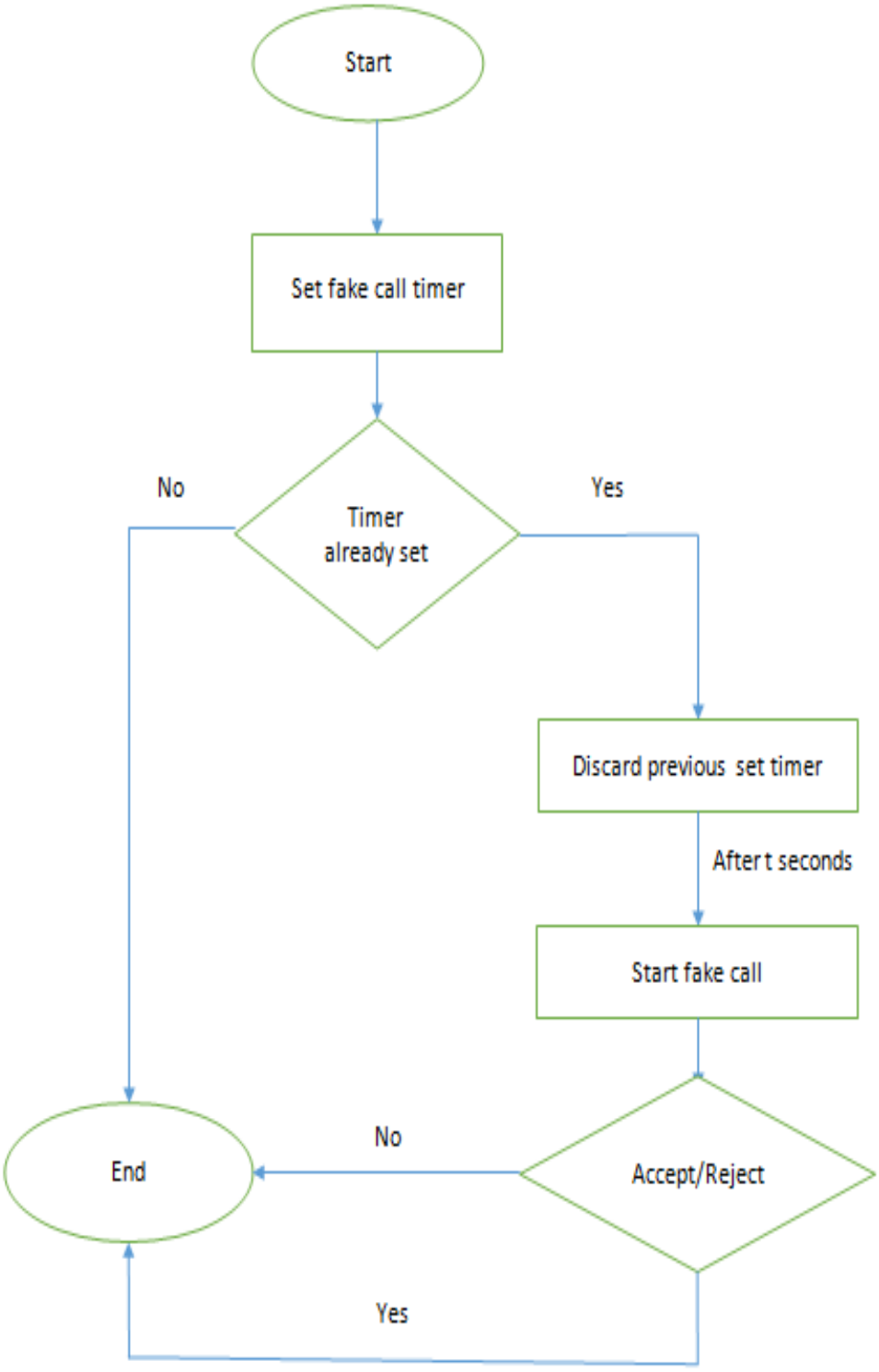


Fig : 7.4 Fake Call Timer

FLOW CHART : TRACK ME

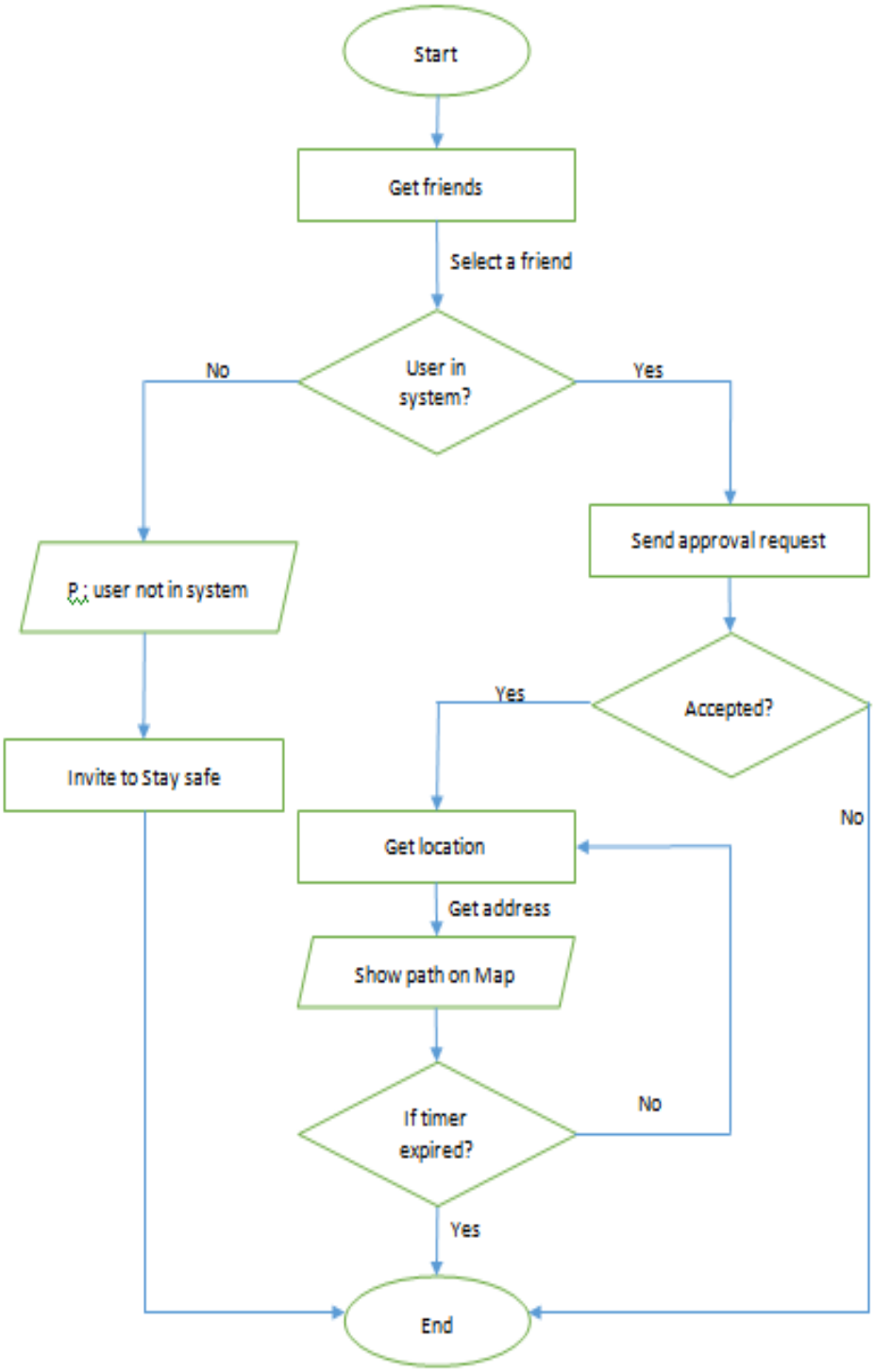


Fig. 7.5 Track me

FLOW CHART : WHERE ARE YOU ?

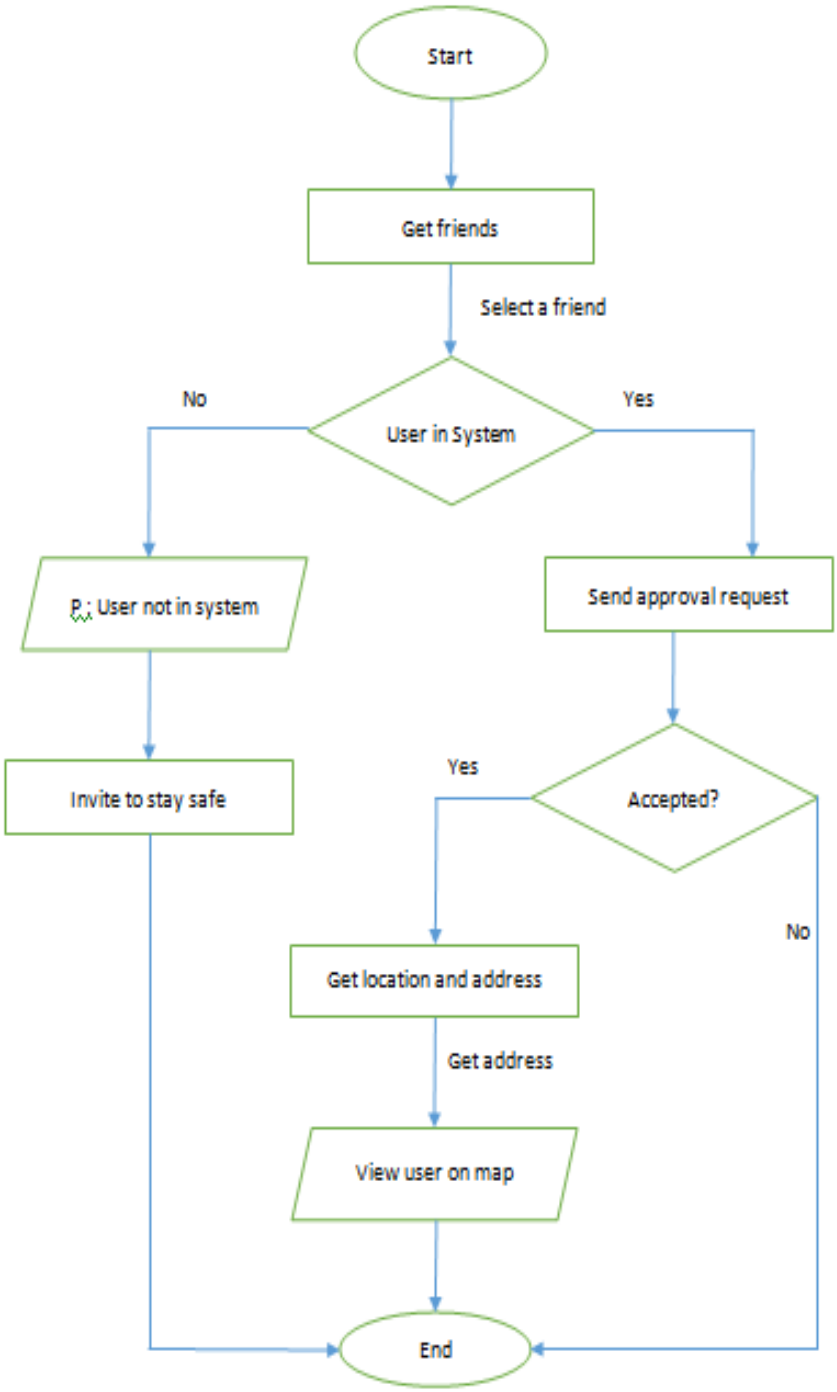


Fig : 7.6 Where Are You

FLOW CHART OF SHAKE FUNCTION : DISTRESS SIGNAL(SOS)

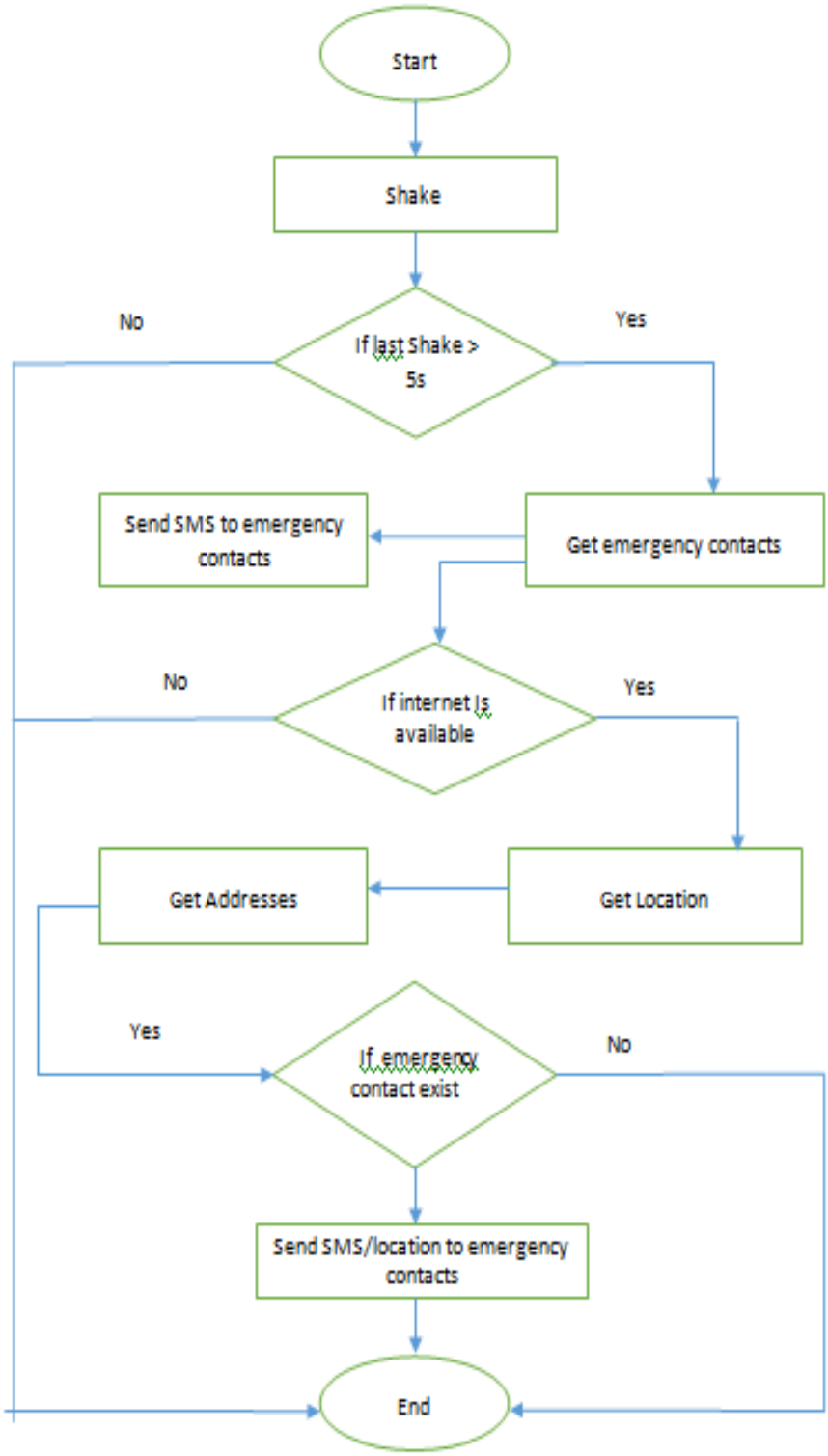


Fig : 7.7 Distress Signal (SOS)

CODING

AccelerometerListener.java

```
package
com.prabhu.womensafetyap  ;


                                public interface AccelerometerListener {

                                  public void onAccelerationChanged(float x, float y, float
                                z);

                                  public void onShake(float force);

                                }
```

AccelerometerManager.java

```
package
com.prabhu.womensaf
etyapp;


                                import java.util.List;
                                import android.content.Context;
                                import android.hardware.Sensor;
                                import android.hardware.SensorEvent;
                                import android.hardware.SensorEventListener;
                                import android.hardware.SensorManager;
                                import android.widget.Toast;

                                public class AccelerometerManager {

                                  private static Context aContext=null;


                                  /** Accuracy configuration */
                                  private static float threshold  = 15.0f;
                                  private static int interval     = 200;
```

```
  private static Sensor sensor;
  private static SensorManager sensorManager;
  // you could use an OrientationListener array instead
  // if you plans to use more than one listener
  private static AccelerometerListener listener;

  /** indicates whether or not Accelerometer Sensor is supported
*/
  private static Boolean supported;
  /** indicates whether or not Accelerometer Sensor is running */
  private static boolean running = false;

  /**
   * Returns true if the manager is listening to orientation changes
   */
  public static boolean isListening() {
     return running;
  }

  /**
   * Unregisters listeners
   */
  public static void stopListening() {
     running = false;
     try {
        if (sensorManager != null && sensorEventListener != null)
{
           sensorManager.unregisterListener(sensorEventListener);
        }
     } catch (Exception e) {}
  }

  /**
   * Returns true if at least one Accelerometer sensor is available
   */
  public static boolean isSupported(Context context) {
     aContext = context;
     if (supported == null) {
        if (aContext != null) {


           sensorManager = (SensorManager) aContext.
```

```
                    getSystemService(Context.SENSOR_SERVICE);

            // Get all sensors in device
            List<Sensor> sensors = sensorManager.getSensorList(
                    Sensor.TYPE_ACCELEROMETER);

            supported = new Boolean(sensors.size() > 0);



        } else {
            supported = Boolean.FALSE;
        }
    }
    return supported;
  }

  /**
   * Configure the listener for shaking
   * @param threshold
   *            minimum acceleration variation for considering
shaking
   * @param interval
   *            minimum interval between to shake events
   */
  public static void configure(int threshold, int interval) {
    AccelerometerManager.threshold = threshold;
    AccelerometerManager.interval = interval;
  }

  /**
   * Registers a listener and start listening
   * @param accelerometerListener
   *            callback for accelerometer events
   */
  public static void startListening( AccelerometerListener
accelerometerListener )
  {

    sensorManager = (SensorManager) aContext.
            getSystemService(Context.SENSOR_SERVICE);

    // Take all sensors in device
```

```
        List<Sensor> sensors = sensorManager.getSensorList(
                Sensor.TYPE_ACCELEROMETER);

        if (sensors.size() > 0) {

            sensor = sensors.get(0);

            // Register Accelerometer Listener
            running = sensorManager.registerListener(
                    sensorEventListener, sensor,
                    SensorManager.SENSOR_DELAY_GAME);

            listener = accelerometerListener;
        }


    }

    /**
     * Configures threshold and interval
     * And registers a listener and start listening
     * @param accelerometerListener
     *          callback for accelerometer events
     * @param threshold
     *          minimum acceleration variation for considering
shaking
     * @param interval
     *          minimum interval between to shake events
     */
    public static void startListening(
            AccelerometerListener accelerometerListener,
            int threshold, int interval) {
        configure(threshold, interval);
        startListening(accelerometerListener);
    }

    /**
     * The listener that listen to events from the accelerometer
listener
     */
    private static SensorEventListener sensorEventListener =
        new SensorEventListener() {
```

```
        private long now = 0;
        private long timeDiff = 0;
        private long lastUpdate = 0;
        private long lastShake = 0;

        private float x = 0;
        private float y = 0;
        private float z = 0;
        private float lastX = 0;
        private float lastY = 0;
        private float lastZ = 0;
        private float force = 0;

        public void onAccuracyChanged(Sensor sensor, int accuracy)
{}

        public void onSensorChanged(SensorEvent event) {
           // use the event timestamp as reference
           // so the manager precision won't depends
           // on the AccelerometerListener implementation
           // processing time
           now = event.timestamp;

           x = event.values[0];
           y = event.values[1];
           z = event.values[2];

           // if not interesting in shake events
           // just remove the whole if then else block
          if (lastUpdate == 0) {
              lastUpdate = now;
              lastShake = now;
              lastX = x;
              lastY = y;
              lastZ = z;
              Toast.makeText(aContext,"No Motion detected",
                Toast.LENGTH_SHORT).show();

             } else {
              timeDiff = now - lastUpdate;

              if (timeDiff > 0) {
```

```
                    /*force = Math.abs(x + y + z - lastX - lastY - lastZ)
                            / timeDiff;*/
                    force = Math.abs(x + y + z - lastX - lastY - lastZ);

                    if (Float.compare(force, threshold) >0 ) {
                       //Toast.makeText(Accelerometer.getContext(),
                       //(now-lastShake)+"  >= "+interval, 1000).show();

                       if (now - lastShake >= interval) {

                          // trigger shake event
                          listener.onShake(force);
                       }
                       else
                       {
                          Toast.makeText(aContext,"No Motion detected.",
                             Toast.LENGTH_SHORT).show();

                       }
                       lastShake = now;
                    }
                    lastX = x;
                    lastY = y;
                    lastZ = z;
                    lastUpdate = now;
                 }
                 else
                 {
                    Toast.makeText(aContext,"No Motion detected",
Toast.LENGTH_SHORT).show();

                 }
              }

              // trigger change event
              listener.onAccelerationChanged(x, y, z);
           }

        };

     }
```

BgService.java

```
package
com.prabhu.
womensafety
app;

import android.annotation.SuppressLint;
import android.app.Service;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.os.Handler;
import android.os.HandlerThread;
import android.os.IBinder;
import android.os.Looper;
import android.os.Message;
import android.telephony.SmsManager;
import android.util.Log;
import android.widget.Toast;

@SuppressLint("HandlerLeak")

public class BgService extends Service implements AccelerometerListener{

  String str_address;
```

```
  private Looper mServiceLooper;
  private ServiceHandler mServiceHandler;

  // Handler that receives messages from the thread.
  private final class ServiceHandler extends Handler {

     public ServiceHandler(Looper looper) {

          super(looper);
     }
   @Override
   public void handleMessage(Message msg) {


          // REPLACE THIS CODE WITH YOUR APP CODE
        // Wait before Toasting Service Message
          // to give the Service Started message time to display.

      // Toast Service Message.
/*                  Context context = getApplicationContext();
                    CharSequence text = "Service Message";
                    int duration = Toast.LENGTH_LONG;
                    Toast toast = Toast.makeText(context, text, duration);
                    toast.show();
*/

      // Service can stop itself using the stopSelf() method.
                    // Not using in this app.  Example statement shown below.
      //stopSelf(msg.arg1);
    }
   }


@Override
public IBinder onBind(Intent arg0) {

          return null;
   }



@Override
public void onCreate() {
```

```
        super.onCreate();


         if (AccelerometerManager.isSupported(this)) {

       AccelerometerManager.startListening(this);
     }
   HandlerThread thread = new
HandlerThread("ServiceStartArguments",android.os.Process.THREAD_PRI
ORITY_BACKGROUND);
   thread.start();

   mServiceLooper = thread.getLooper();

   mServiceHandler = new ServiceHandler(mServiceLooper);
}


@Override
public int onStartCommand(Intent intent, int flags, int startId) {

        // Get message from message pool using handler.
   Message msg = mServiceHandler.obtainMessage();

   // Set start ID (unique to the specific start) in message.
   msg.arg1 = startId;

   // Send message to start job.
   mServiceHandler.sendMessage(msg);

   // Toast Service Started message.
//         Context context = getApplicationContext();




/*         CharSequence text = "Service Started";
           int duration = Toast.LENGTH_SHORT;
           Toast toast = Toast.makeText(context, text, duration);
           toast.show();
  */
```

```
            // Start a sticky.
            return START_STICKY;
}



public class GeocoderHandler extends Handler {
    @Override
    public void handleMessage(Message message) {

            Toast.makeText(getApplicationContext(), "geocoderhandler
started", Toast.LENGTH_SHORT).show();



        switch (message.what) {
           case 1:
              Bundle bundle = message.getData();
              str_address = bundle.getString("address");
             // TelephonyManager
tmgr=(TelephonyManager)BgService.this.getSystemService(Context.TELEP
HONY_SERVICE);
             //  String ph_number=tmgr.getLine1Number();
            SQLiteDatabase db;
                        db=openOrCreateDatabase("NumDB",
Context.MODE_PRIVATE, null);
                        Cursor c=db.rawQuery("SELECT * FROM details",
null);
                        Cursor c1=db.rawQuery("SELECT * FROM SOURCE",
null);

              String source_ph_number=c1.getString(0);
               while(c.moveToNext())
                        {
                           String target_ph_number=c.getString(1);

        //        SmsManager smsManager=SmsManager.getDefault();
        //        smsManager.sendTextMessage("+918121662586",
"+918121662586", "Please help me. I need help immediately. This is where i
am now:"+str_address, null, null);

                        Toast.makeText(getApplicationContext(),
"Source:"+source_ph_number+"Target:"+target_ph_number,
Toast.LENGTH_SHORT).show();
```

```
                    }
                db.close();

                break;
            default:
            str_address = null;
        }
            Toast.makeText(getApplicationContext(), str_address,
Toast.LENGTH_SHORT).show();

    }
  }



@Override
public void onAccelerationChanged(float x, float y, float z) {
        // TODO Auto-generated method stub

}
@Override
public void onShake(float force) {

        GPSTracker gps;
        gps = new GPSTracker(BgService.this);
    if(gps.canGetLocation()){

        double latitude = gps.getLatitude();
        double longitude = gps.getLongitude();

        RGeocoder RGeocoder = new RGeocoder();
        RGeocoder.getAddressFromLocation(latitude,
longitude,getApplicationContext(), new GeocoderHandler());
        Toast.makeText(getApplicationContext(), "onShake",
Toast.LENGTH_SHORT).show();


    }
    else{
        gps.showSettingsAlert();
        }
```

```
}



// onDestroy method.   Display toast that service has stopped.
@Override
public void onDestroy() {
        super.onDestroy();

        // Toast Service Stopped.
        Context context = getApplicationContext();

          Log.i("Sensor", "Service  distroy");

    if (AccelerometerManager.isListening()) {

       AccelerometerManager.stopListening();

    }

        CharSequence text = "Women Safety App Service Stopped";
        int duration = Toast.LENGTH_SHORT;
        Toast toast = Toast.makeText(context, text, duration);
        toast.show();


}



}
```

Display.java

```
package
com.prabhu.womensafety
app;
```

```
import android.app.Activity;
import android.app.AlertDialog.Builder;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;


public class Display extends Activity{



Cursor c;
@Override
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_display);


        SQLiteDatabase db;
        db=openOrCreateDatabase("NumDB",
Context.MODE_PRIVATE, null);

          c=db.rawQuery("SELECT * FROM details", null);
          if(c.getCount()==0)
          {
             showMessage("Error", "No records found.");
             return;
          }
          StringBuffer buffer=new StringBuffer();
          while(c.moveToNext())
          {
             buffer.append("Name: "+c.getString(0)+"\n");
             buffer.append("Number: "+c.getString(1)+"\n");
          }
          showMessage("Details", buffer.toString());
        Intent i_startservice=new
Intent(Display.this,BgService.class);
```

```
                startService(i_startservice);


        }

        public void showMessage(String title,String message)
          {
             Builder builder=new Builder(this);
             builder.setCancelable(true);
             builder.setTitle(title);
             builder.setMessage(message);
             builder.show();
          }




        public void back(View v) {
                Intent i_back=new
        Intent(Display.this,MainActivity.class);
                startActivity(i_back);

                }



        }
```

Register.java

```
  package
  com.prabhu.w
  omensafetyapp
  ;


                import android.app.Activity;
                import android.content.Context;
```

```
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;


public class Register extends Activity {

EditText name,number;




@Override
protected void onCreate(Bundle savedInstanceState) {
	super.onCreate(savedInstanceState);
	setContentView(R.layout.activity_register);
	//Toast.makeText(getApplicationContext(), "Activity
created",Toast.LENGTH_LONG).show();


}



public void display(View v) {
	Intent i_view=new Intent(Register.this,Display.class);
	startActivity(i_view);

	}

public void instructions(View v) {
	Intent i_help=new Intent(Register.this,Instructions.class);
	startActivity(i_help);

	}

public void storeInDB(View v) {
	Toast.makeText(getApplicationContext(), "save
started",Toast.LENGTH_LONG).show();
```

```
        name = (EditText) this.findViewById(R.id.editText1);
        number = (EditText) this.findViewById(R.id.editText2);
        String str_name=name.getText().toString();
        String str_number=number.getText().toString();
        SQLiteDatabase db;
        db=openOrCreateDatabase("NumDB",
Context.MODE_PRIVATE, null);
        //Toast.makeText(getApplicationContext(), "db
created",Toast.LENGTH_LONG).show();

        db.execSQL("CREATE TABLE IF NOT EXISTS details(name
VARCHAR,number VARCHAR);");
        //Toast.makeText(getApplicationContext(), "table
created",Toast.LENGTH_LONG).show();

        Cursor c=db.rawQuery("SELECT * FROM details", null);
          if(c.getCount()<2)
          {
                db.execSQL("INSERT INTO details
VALUES('"+str_name+"','"+str_number+"');");
                Toast.makeText(getApplicationContext(),
"Successfully Saved",Toast.LENGTH_SHORT).show();
          }
          else {

                db.execSQL("INSERT INTO details
VALUES('"+str_name+"','"+str_number+"');");
                Toast.makeText(getApplicationContext(), "Maximun
Numbers limited reached. Previous numbers are
replaced.",Toast.LENGTH_SHORT).show();
          }


        db.close();

        }

}
```

Verify.java

```
package
com.prabhu.wome
nsafetyapp;


                import android.os.Bundle;
                import android.app.Activity;
                import android.view.Menu;
                import android.view.MenuItem;
                import android.view.View;
                import android.widget.EditText;
                import android.widget.Toast;
                import android.support.v4.app.NavUtils;
                import android.annotation.TargetApi;
                import android.content.Context;
                import android.content.Intent;
                import android.database.sqlite.SQLiteDatabase;
                import android.os.Build;


                public class Verify extends Activity {


                @Override
                protected void onCreate(Bundle savedInstanceState) {
                        super.onCreate(savedInstanceState);
                        setContentView(R.layout.activity_verify);
                        // Show the Up button in the action bar.
                        setupActionBar();



                }


                public void verify_no(View v) {
                        EditText source_no = (EditText)
                this.findViewById(R.id.editText1);
                        String str_source_no=source_no.getText().toString();
                        SQLiteDatabase db;
                        db=openOrCreateDatabase("NumDB",
                Context.MODE_PRIVATE, null);
                //      if(source_no.getText()!=null){
```

```
		db.execSQL("CREATE TABLE IF NOT EXISTS
source(number VARCHAR);");
		db.execSQL("INSERT INTO source
VALUES('"+str_source_no+"');");
		Toast.makeText(getApplicationContext(), str_source_no+"
Successfully Saved",Toast.LENGTH_SHORT).show();
		db.close();
		back(v);
//			}
//			else{
//				Toast.makeText(getApplicationContext(),
"Enter Your Number.",Toast.LENGTH_SHORT).show();
//		}
}
```

```
/**
 * Set up the {@link android.app.ActionBar}, if the API is available.
 */
@TargetApi(Build.VERSION_CODES.HONEYCOMB)
private void setupActionBar() {
		if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.HONEYCOMB) {

		getActionBar().setDisplayHomeAsUpEnabled(true);
		}
}


@Override
public boolean onCreateOptionsMenu(Menu menu) {
		// Inflate the menu; this adds items to the action bar if it is
present.
		getMenuInflater().inflate(R.menu.verify, menu);
		return true;
}
```

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case android.R.id.home:
                // This ID represents the Home or Up button. In the
case of this
                // activity, the Up button is shown. Use NavUtils to
allow users
                // to navigate up one level in the application
structure. For
                // more details, see the Navigation pattern on
Android Design:
                //
                //
http://developer.android.com/design/patterns/navigation.html#up-vs-
back
                //
                NavUtils.navigateUpFromSameTask(this);
                return true;
        }
        return super.onOptionsItemSelected(item);
}


public void back(View v) {
        Intent i_back=new Intent(Verify.this,MainActivity.class);
        startActivity(i_back);

        }


}
```

MainActivity.java

```
package
com.prabhu.womensafety
app;
```

```
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;


public class MainActivity extends Activity {


@Override
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
}



public void register(View v) {
        Intent i_register=new
Intent(MainActivity.this,Register.class);
        startActivity(i_register);

}

public void display_no(View v) {
Intent i_view=new Intent(MainActivity.this,Display.class);
startActivity(i_view);

}


public void instruct(View v) {
Intent i_help=new Intent(MainActivity.this,Instructions.class);
  startActivity(i_help);
}


public void verify(View v) {
Intent i_verify=new Intent(MainActivity.this,Verify.class);
  startActivity(i_verify);
}
}
```

RGeocoder.java

```
package
com.prabhu.womensafetyapp;

import java.io.IOException;
import java.util.List;
import java.util.Locale;

import android.content.Context;
import android.location.Address;
import android.location.Geocoder;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;

public class RGeocoder {

   private static final String TAG = "LocationAddress";

        public void getAddressFromLocation(final double
latitude, final double longitude,
        final Context context, final Handler handler) {

                Thread thread = new Thread() {
        @Override
        public void run() {
           Geocoder geocoder = new Geocoder(context,
Locale.getDefault());
           String result = null;
           try {

              List<Address> addressList =
geocoder.getFromLocation(latitude, longitude, 1);
              if (addressList != null && addressList.size() >
0) {
                 Address address = addressList.get(0);
                 StringBuilder sb = new StringBuilder();
                 for (int i = 0; i <
address.getMaxAddressLineIndex(); i++) {
```

```
sb.append(address.getAddressLine(i)).append("\n");
                    }

sb.append(address.getLocality()).append("\n");

sb.append(address.getPostalCode()).append("\n");
                    sb.append(address.getCountryName());
                    result = sb.toString();



                }
            }


            catch (IOException e) {
                Log.e(TAG, "Unable connect to Geocoder", e);
            }


            finally {
                Message message = Message.obtain();
                message.setTarget(handler);
                if (result != null) {
                    message.what = 1;
                    Bundle bundle = new Bundle();
                    result = "Latitude: " + latitude + " Longitude:
" + longitude +
                            "\n\nAddress:\n" + result;
                    bundle.putString("address", result);
                    message.setData(bundle);
                } else {
                    message.what = 1;
                    Bundle bundle = new Bundle();
                    result = "Latitude: " + latitude + " Longitude:
" + longitude +
                            "\n Unable to get address for this lat-
long.";
                    bundle.putString("address", result);
                    message.setData(bundle);
                }
                message.sendToTarget();
            }
        }
    };
```

```
                thread.start();




            }
        }
```

Instructions.java

```
package
com.prabhu.womensafetyapp;


import android.os.Bundle;
import android.view.View;
import android.app.Activity;
import android.content.Intent;


public class Instructions extends Activity {


    @Override
    protected void onCreate(Bundle savedInstanceState)
{
        super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_instructions);
        }

    public void back(View v) {
        Intent i_back=new
Intent(Instructions.this,MainActivity.class);
        startActivity(i_back);

        }




}
```

GPSTracker.java

```
package
com.prabhu.womensafet
yapp;

import android.app.AlertDialog;
import android.app.Service;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.os.IBinder;
import android.provider.Settings;
import android.util.Log;


public class GPSTracker extends Service implements
LocationListener {


    private final Context mContext;


    // flag for GPS status
    boolean isGPSEnabled = false;


    // flag for network status
    boolean isNetworkEnabled = false;


    // flag for GPS status
    boolean canGetLocation = false;


    Location location; // location
    double latitude; // latitude
    double longitude; // longitude
```

```
	// The minimum distance to change Updates in meters
	private static final long
MIN_DISTANCE_CHANGE_FOR_UPDATES = 10; // 10
meters


	// The minimum time between updates in milliseconds
	private static final long MIN_TIME_BW_UPDATES =
1000 * 60 * 1; // 1 minute


	// Declaring a Location Manager
	protected LocationManager locationManager;


	public GPSTracker(Context context) {
		this.mContext = context;
		getLocation();
	}


	public Location getLocation() {
		try {
			locationManager = (LocationManager)
mContext

	.getSystemService(LOCATION_SERVICE);


				// getting GPS status
				isGPSEnabled = locationManager

	.isProviderEnabled(LocationManager.GPS_PROVIDER);


				// getting network status
				isNetworkEnabled = locationManager

	.isProviderEnabled(LocationManager.NETWORK_PRO
VIDER);


				if (!isGPSEnabled &&
!isNetworkEnabled) {
```

```
                        // no network provider is enabled
                    } else {
                        this.canGetLocation = true;
                        if (isNetworkEnabled) {

            locationManager.requestLocationUpdates(

            LocationManager.NETWORK_PROVIDER,

            MIN_TIME_BW_UPDATES,

            MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
                                Log.d("Network",
"Network");
                                if (locationManager != null)
{
                                    location =
locationManager

            .getLastKnownLocation(LocationManager.NETWORK_
PROVIDER);
                                    if (location != null)
{
                                        latitude =
location.getLatitude();
                                        longitude =
location.getLongitude();
                                    }
                                }
                            }
                            // if GPS Enabled get lat/long using
GPS Services
                            if (isGPSEnabled) {
                                if (location == null) {

            locationManager.requestLocationUpdates(

            LocationManager.GPS_PROVIDER,

            MIN_TIME_BW_UPDATES,

            MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
                                    Log.d("GPS
Enabled", "GPS Enabled");
                                    if (locationManager
!= null) {
```

```
                                                location =
locationManager

        .getLastKnownLocation(LocationManager.GPS_PROVI
DER);
                                                if (location
!= null) {

        latitude = location.getLatitude();

        longitude = location.getLongitude();
                                                }
                                            }
                                        }
                                    }
                                }


                } catch (Exception e) {
                        e.printStackTrace();
                }


                return location;
        }

        /**
         * Stop using GPS listener
         * Calling this function will stop using GPS in your app
         * */
        public void stopUsingGPS(){
                if(locationManager != null){

        locationManager.removeUpdates(GPSTracker.this);
                }
        }

        /**
         * Function to get latitude
         * */
        public double getLatitude(){
                if(location != null){
                        latitude = location.getLatitude();
                }

                // return latitude
```

```
                return latitude;
        }

        /**
         * Function to get longitude
         * */
        public double getLongitude(){
                if(location != null){
                        longitude = location.getLongitude();
                }

                // return longitude
                return longitude;
        }

        /**
         * Function to check GPS/wifi enabled
         * @return boolean
         * */
        public boolean canGetLocation() {
                return this.canGetLocation;
        }

        /**
         * Function to show settings alert dialog
         * On pressing Settings button will lauch Settings Options
         * */
        public void showSettingsAlert(){
                AlertDialog.Builder alertDialog = new
AlertDialog.Builder(mContext);

      // Setting Dialog Title
      alertDialog.setTitle("GPS is settings");

      // Setting Dialog Message
      alertDialog.setMessage("GPS is not enabled. Do you want
to go to settings menu?");

      // On pressing Settings button
      alertDialog.setPositiveButton("Settings", new
DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog,int which) {
                Intent intent = new
Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                mContext.startActivity(intent);
        }
```

```
        });

        // on pressing cancel button
        alertDialog.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
            }
        });

        // Showing Alert Message
        alertDialog.show();
            }


            @Override
            public void onLocationChanged(Location location) {
            }


            @Override
            public void onProviderDisabled(String provider) {
            }


            @Override
            public void onProviderEnabled(String provider) {
            }


            @Override
            public void onStatusChanged(String provider, int status,
Bundle extras) {
            }


            @Override
            public IBinder onBind(Intent arg0) {
                    return null;
            }


}
```

CHAPTER-8

TESTING

During testing the programs to be tested are executed with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as expected. Testing forms is the first in determining errors in the program. Once programs were tested individually then the system as a whole was tested. During testing the system is used experimentally to ensure that the software does not fail i.e. it will run according to its specification. The program executed to check for any syntax and logical errors. The errors are corrected and a test is made to determine whether the program is doing what it is supposed to do.

There are generally four recognized levels of tests :

- Unit Testing
- Integration Testing
- System Testing
- Acceptance Testing

Unit testing : Testing of individual software components or modules. Typically done by the programmer or not by testers, as it requires detailed knowledge of the internal program design and code.

Integration testing : Testing of integrated modules to verify combined functionality after integration. Modules are typically code modules, individual applications, client and server applications on a network etc. This type of testing is especially relevant to client/server and distributed systems.

System Testing : System testing, or end-to-end testing, tests a completely integrated system to verify that it meets its requirements. Software testing should ensure that the program, as well as working as expected, does not also destroy or partially corrupt its operating environment or cause other processes within that environment to become inoperative (this includes not corrupting shared

memory, not consuming or locking up excessive resources and leaving any parallel processes unharmed by its presence).

Acceptance Testing : Normally this type of testing is done to verify if a system meets the users specified requirements. Users or customers do this testing to determine whether to accept an application.

8.1 FUNCTIONAL TESTING

- The identification of functions that the software is expected to perform.
- The creation of input data based on the function's specifications.
- The determination of output based on the function's specifications.
- The comparison of actual and expected outputs.

8.1.1 IDENTIFICATION OF FUNCTIONS

- Login
- Distress Signal (SOS)
- Track Me
- Where Are You ?
- Scream Function
- Fake Caller
- Logout

8.2 STRUCTURAL TESTING

Structural test design techniques includes :

- Control flow Testing: - Whether the low of control of the code is in order i.e., level wise
- Data flow Testing: - When data flow between two blocks or within a block occurs. Is it running as needed or if there are any bugs or presents?
- Branch Testing: - The test of branches and loops of the code is done.
- Path Testing: - It can test paths within a unit, paths between units during integration and between subsystems.

This type of testing is also called white box testing. Here, we check the code internally for flaws and bugs.

8.3 LEVELS OF TESTING

There are different levels of testing as follows :

Alpha Testing : There are three types of alpha testing namely -

- Unit Testing.
- Integration Testing (Top Down & Bottom Up)
- System Testing.

Acceptance Testing : Acceptance Testing is a formal testing conducted to determine whether a system satisfies its acceptance criteria.

There are two categories of acceptance testing :

- User Acceptance Testing
- Business Acceptance Testing

Beta Testing : It is also known as field testing. It is the second phase of software testing in which a sampling of the intended audience tries the product out.

Goal of the beta testing is to place the application in the hands of real users in order to discover any flaws or issues from the user's perspective.

CHAPTER-9

IMPLEMENTATION

9.1 IMPLEMENTATION OF THE PROJECT

There are three types of implementations :-

- Implementation of android system to replace a manual system.
- Implementation of a new android system to replace an existing system.
- Implementation of a modified application to replace an existing one, using the same system.

9.2 IMPLEMENTATION AND SOFTWARE MAINTENANCE

9.2.1 POST IMPLEMENTATION

After the implementation, the beta version of the application would be available and ready to use for all the customers. During this phase, we try to identify any bugs that may have crept into the system despite all the previous testing done. This is also called beta testing during the Post.

9.2.2 SOFTWARE MAINTENANCE

Maintenance is the enigma of system development. It holds the software industry captive, tying up programming resources. Analysts and programmers spend more time maintaining programs than writing them. Maintenance is not considered a part of software development, its activity an extremely important part of the life of a software product.

- Corrective Maintenance: - After the Implementation, Correcting the residual errors if any. If such errors are discovered, the source of it should be detected and removed. This phenomenon falls under corrective maintenance.
- Perfective Maintenance: - Sometimes changes have to be done according to the user requirements. This type of changes to the software is called perfective maintenance.
- Adaptive Maintenance: - Software often must be upgraded and enhanced to include more features and provide more services. This also requires modification of software.

CHAPTER-10

PROJECT LEGACY

10.1 CURRENT STATUS OF THE PROJECT

The user could generate a distress signal (SOS) through his/her end and a push notification will be sent to the emergency contacts of the user with the exact location and respective SMS will also be sent at the end user. Currently the "Stay Safe Application" is in the testing phase and publishing applications to the Google Play Store will be implemented in May.

10.2 REMAINING AREAS OF CONCERN

Although this application has all the features and details of all the emergencies modules that would help the user to escape from the unwanted situations. i.e threat and boring social events, but it still needs further modifications. These modifications will be done later.

Some of the features are :

- **Widgets :** To make access to applications.
- **Location Rating :** The user would be able to see the feedback rating while passing through a street or place as per done by the survey. The user would also be able to view the nearest police station and hospital location for emergency purposes.

10.3 TECHNICAL AND MANAGERIAL LESSONS LEARNT

It has helped us to sharpen our knowledge and skills, develop better appreciation of practical problems of application development and to apply the concepts and techniques to developmental problems. This experience is going to help us immensely in further learning of advanced concepts in android application development and to plan our career in the light of practical experience now. We have examples to relate and it will facilitate better and easier learning for us.

Technical lessons learnt

- Installation and knowledge of Android Studio.

- Android API levels and their corresponding changes.
- Designing of the system.
- Working with Sensors in order to generate Distress Signal (SOS).
- Designing custom interface using xml files.
- Various services of Android to make a call.
- Publishing Application to the Google Play Store.

Managerial lessons learnt

- Ensuring quality and integrity of data.
- Planning of Duration and schedule of the project.
- Strategic planning to avoid miscommunication among the team members.
- Participative Leadership.
- Coordination.
- Risk Analysis and prevention.
- Integrating individual work to make it collaborative work.
- Defining smaller goals to achieve a bigger common goal.

CHAPTER-11

USER MANUAL

When a user launches the application in his/her Android phone, the very first screen which lands is the Login Screen. First the user has to register himself by entering the details as the respective name and contact number of the user.

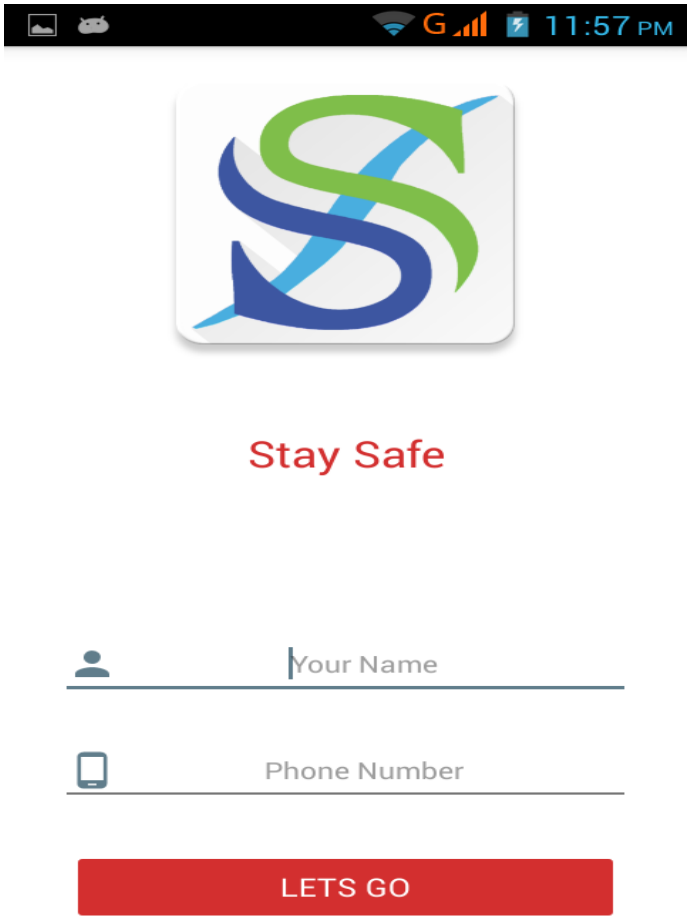


Fig 11.1 : Login Page Interface

After entering the correct details in order to Sign Up, the confirmation code (OTP) will be sent to the user at his/her respective contact number.

Fig 11.2 : Verification through OTP Interface

After successfully logged in by the user, main application pop up window will open up which consists of the following functions :-

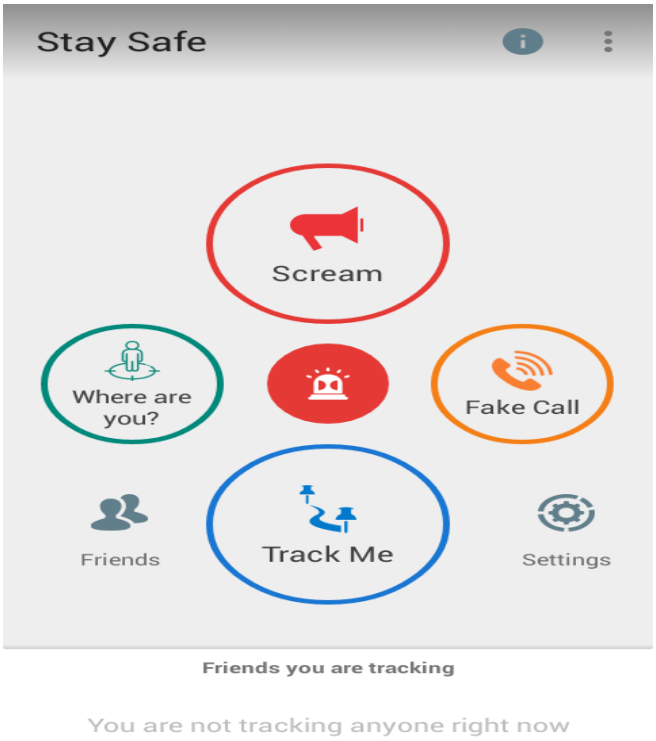


Fig 11.3 Main Interface of The Application

- **Scream Function :** The Scream function will allow the user to generate a distraction in order to escape from the unsafe situation.

The user could also select the type of scream as per the requirement from the "Settings" icon.

- **Fake Call :** The fake call timer allows the user to make fake calls in the time of need. It helps users to escape from an undesirable situation by citing an important call from anyone who needs him/her urgently. After a long term press on the icon will also start a fake call for the user.

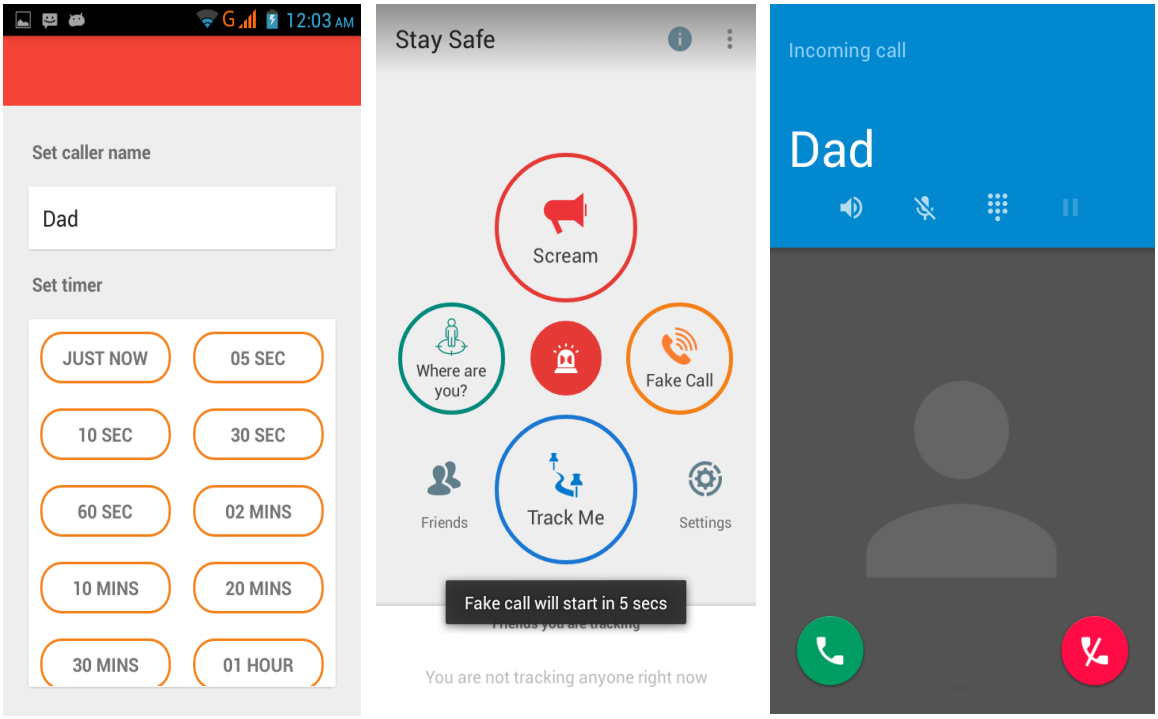


Fig 11.4 : Fake Caller Interface

Where are you ? : The where are you feature allows to view the static location of the user and SMS will be sent at the receiver end with the exact static location of the user.

After selecting the Where Are You icon, users have to pick a friend from the friends list and the Where are you request will be sent at the receiver end. The receiver will accept the request and the location will be sent at the user end.

Fig 11.5 Interface of Where Are You ?

Track Me : The track me feature allows the user to view the exact dynamic location of the victim. First users have to send the Track Me request at the receiver's end. The receiver will accept the request and then his/her name will appear on the friends you are tracking on the bottom of the application. The user could select that friend from there and then it will get

automatically redirected to the Google maps from where the user could view the exact location of the victim and also where he/she is heading to.

Fig 11.6 : Track Me Interface



Friends : The friend list shows the list of the friend's with whom the user is connected to. The user could add a friend by selecting the "Add a friend" icon on the bottom right corner. The user could add any contact no. directly or could also import that from the "Contacts".

Distress Signal (SOS) : The distress signal will be generated by the user in case of an emergency. In order to generate the distress signal the user has to shake up his/her phone, then a distress signal will appear at the user end. The default timer for sending this signal is 5 sec. The default timer is set as If the user wants to discard the signal from his/her end. In the end a distress signal will be sent to the emergency contacts with the exact location of the victim. A push notification will also be sent at the user end having all the details.



Fig 11.8 : Distress Signal (SOS) Interface

Settings : This consists of the following features :

- Emergency Services : It allows the Stay Safe Application to send emergency notifications and SMS with the exact location to the emergency contacts.
- Low Battery Alert : The low battery alert feature allows the Stay Safe Application to send low battery alerts and SMS to the emergency contacts.
- Set Scream Sound : The user could select any scream sound as per the requirement.
- Fake Call Timer (On Long press) : The user could set the fake call default timer as per the requirement.

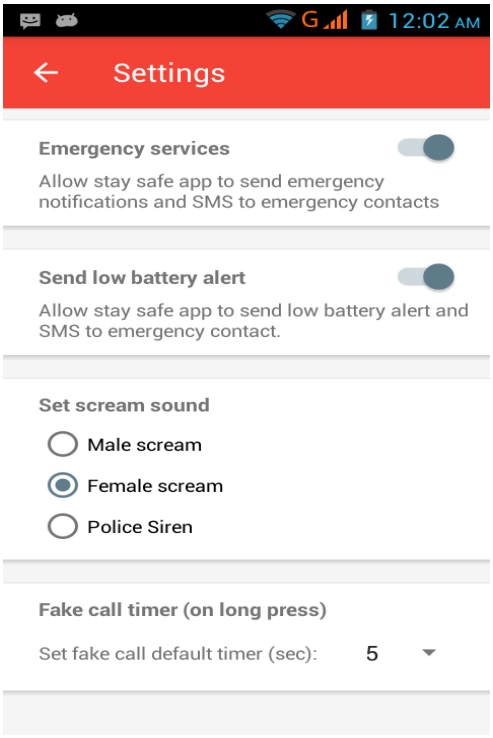


Fig 11.9 : Settings Interface

Logout : The user could logout from the application by selecting the "Menu" tab on the top right corner of the application

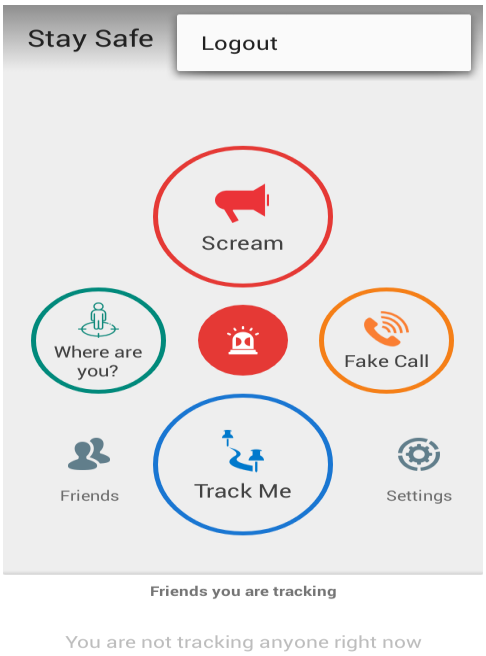


Fig 11.10 : Logout Interface

12. BIBLIOGRAPHY

- www.android.com/intl/en_in/
- en.wikipedia.org/wiki/Android_(operating_system)
- developer.android.com/studio/index.html
- tutorialspoint.com/android/android_studio.htm
- android-developers.blogspot.com/2016/09/android-studio-2-2.html
- www.parallelcodes.com/android-game-tutorial/
- www.developerfeed.com/Android/

- Hello, Android: Introducing Google's Mobile Development Platform Book by Ed Burnette
- Android Programming: The Big Nerd Ranch Guide Book by Brian Hardy
- Android Recipes: A Problem-Solution Approach Book by Dave Smith
- Programming Android Book by G. Blake Meike, Laird Dornin, Masumi Nakamura, and Zigurd R. Mednieks.

References

1. Kamal Acharya. School management system project report. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172254873.34023165/v1

2. Kamal Acharya. A CASE STUDY OF CINEMA MANAGEMENT SYSTEM PROJECT. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172254873.30191075/v1

3. Kamal Acharya. A CASE STUDY ON ONLINE TICKET BOOKING SYSTEM PROJECT. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172254872.26972790/v1

4. Kamal Acharya. Web chatting application project report management system. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172254871.18588592/v1

5. Kamal Acharya. RETAIL STORE MANAGEMENT SYSTEM PROJECT REPORT. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172254871.14590154/v1

6. Kamal Acharya. SUPERMARKET MANAGEMENT SYSTEM PROJECT REPORT. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172252491.19145062/v1

7. Kamal Acharya. SOCIAL MEDIA MANAGEMENT SYSTEM PROJECT REPORT. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172252491.11210579/v1

8. Kamal Acharya. Online music portal management system project report. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172252488.89734698/v1

9. Kamal Acharya. COLLEGE BUS MANAGEMENT SYSTEM PROJECT REPORT. Authorea. July 31, 2024. DOI: https://doi.org/10.22541/au.172245277.70798942/v1

10. Kamal Acharya. AUTOMOBILE MANAGEMENT SYSTEM PROJECT REPORT. Authorea. July 31, 2024. DOI: https://doi.org/10.22541/au.172245276.67982593/v1

11. Kamal Acharya. Ludo management system project report. Authorea. July 31, 2024. DOI: https://doi.org/10.22541/au.172243999.98091616/v1

12. Kamal Acharya. Literature online quiz system project report. Authorea. July 31, 2024 DOI: https://doi.org/10.22541/au.172243825.53562953/v1

13. Kamal Acharya. Avoid waste management system project. Authorea. July 29, 2024. DOI: https://doi.org/10.22541/au.172228528.85022205/v1

14. Kamal Acharya. CHAT APPLICATION THROUGH CLIENT SERVER MANAGEMENT SYSTEM PROJECT. Authorea. July 29, 2024. DOI: https://doi.org/10.22541/au.172228527.74316529/v1

15. Acharya, Kamal, Online Job Portal Management System (May 5, 2024). Available at SSRN: https://ssrn.com/abstract=4817534 or *http://dx.doi.org/10.2139/ssrn.4817534*

16. Acharya, Kamal, Employee leave management system. (May 7, 2024). Available at SSRN: https://ssrn.com/abstract=4819626 or *http://dx.doi.org/10.2139/ssrn.4819626*

17. Acharya, Kamal, Online electricity billing project report. (May 7, 2024). Available at SSRN: https://ssrn.com/abstract=4819630 or *http://dx.doi.org/10.2139/ssrn.4819630*

18. Acharya, Kamal, POLICY MANAGEMENT SYSTEM PROJECT REPORT. (December 10, 2023). Available at SSRN: https://ssrn.com/abstract=4831694 or http://dx.doi.org/10.2139/ssrn.4831694

19. Acharya, Kamal, Software testing for project report. (May 16, 2023). Available at SSRN: https://ssrn.com/abstract=4831028 or http://dx.doi.org/10.2139/ssrn.4831028

20. Acharya, Kamal, ONLINE CRIME REPORTING SYSTEM PROJECT. (August 10, 2022). Available at SSRN: https://ssrn.com/abstract=4831015 or http://dx.doi.org/10.2139/ssrn.4831015

21. Acharya, Kamal, Burger ordering system project report. (October 10, 2022). Available at SSRN: https://ssrn.com/abstract=4832704 or http://dx.doi.org/10.2139/ssrn.4832704

23. Acharya, Kamal, Teachers Record Management System Project Report (December 10, 2023). Available at SSRN: https://ssrn.com/abstract=4833821 or http://dx.doi.org/10.2139/ssrn.4833821

23. Acharya, Kamal, Dairy Management System Project Report (December 20, 2020). Available at SSRN: https://ssrn.com/abstract=4835231 or http://dx.doi.org/10.2139/ssrn.4835231

24. Acharya, Kamal, Electrical Shop Management System Project (December 10, 2019). Available at SSRN: https://ssrn.com/abstract=4835238 or http://dx.doi.org/10.2139/ssrn.4835238

25. Acharya, Kamal, Online book store management system project report. (Febuary 10, 2020). Available at SSRN: https://ssrn.com/abstract=4835277 or http://dx.doi.org/10.2139/ssrn.4835277

26. Acharya, Kamal, Paint shop management system project report. (January 10, 2019). Available at SSRN: https://ssrn.com/abstract=4835441 or http://dx.doi.org/10.2139/ssrn.4835441

27. Acharya, Kamal, Supermarket billing system project report. (August 10, 2021). Available at SSRN: https://ssrn.com/abstract=4835474 or http://dx.doi.org/10.2139/ssrn.4835474

28. Acharya, Kamal, Online taxi booking system project report. (March 10, 2022). Available at SSRN: https://ssrn.com/abstract=4837729 or http://dx.doi.org/10.2139/ssrn.4837729

29. Acharya, Kamal, Online car servicing system project report. (March 10, 2023). Available at SSRN: https://ssrn.com/abstract=4837832 or http://dx.doi.org/10.2139/ssrn.4837832

30. Acharya, Kamal, School management system project report. (July 10, 2021). Available at SSRN: https://ssrn.com/abstract=4837837 or http://dx.doi.org/10.2139/ssrn.4837837

31. Acharya, Kamal, Furniture Showroom Management System Project Report (March 21, 2021). Available at SSRN: https://ssrn.com/abstract=4839422 or http://dx.doi.org/10.2139/ssrn.4839422

32. Acharya, Kamal, Online Vehicle Rental System Project Report (March 21, 2019). Available at SSRN: https://ssrn.com/abstract=4839429 or http://dx.doi.org/10.2139/ssrn.4839429

33. Acharya, Kamal, Fruit Shop Management System Project Report (August 10, 2023). Available at SSRN: https://ssrn.com/abstract=4841048 or http://dx.doi.org/10.2139/ssrn.4841048

34. Acharya, Kamal, Hall Booking Management System Project Report (December 21, 2023). Available at SSRN: https://ssrn.com/abstract=4841055 or http://dx.doi.org/10.2139/ssrn.4841055

35. Acharya, Kamal, Lundry Management System Project Report (October 21, 2023). Available at SSRN: https://ssrn.com/abstract=4841059 or http://dx.doi.org/10.2139/ssrn.4841059

36. *Acharya, Kamal, A CASE STUDY OF CINEMA MANAGEMENT SYSTEM PROJECT (September 25, 2023). Available at SSRN: https://ssrn.com/abstract=4841209* or *http://dx.doi.org/10.2139/ssrn.4841209*

37. *Acharya, Kamal, A CASE STUDY ON ONLINE TICKET BOOKING SYSTEM PROJECT (May 25, 2024). Available at SSRN: https://ssrn.com/abstract=4841210* or *http://dx.doi.org/10.2139/ssrn.4841210*

38. Acharya, Kamal, ONLINE DATING MANAGEMENT SYSTEM PROJECT REPORT. (April 25, 2023). Available at SSRN: https://ssrn.com/abstract=4842066 or http://dx.doi.org/10.2139/ssrn.4842066

39. Acharya, Kamal, ONLINE RESUME BUILDER MANAGEMENT SYSTEM PROJECT REPORT. (April 25, 2021). Available at SSRN: https://ssrn.com/abstract=4842071 or http://dx.doi.org/10.2139/ssrn.4842071

40. *Acharya, Kamal, TOLL TEX MANAGEMENT SYSTEM PROJECT REPORT (August 21, 2023). Available at SSRN: https://ssrn.com/abstract=4842082 or http://dx.doi.org/10.2139/ssrn.4842082*

41. *Acharya, Kamal, Chat Application Through Client Server Management System Project Report (June 25, 2023). Available at SSRN: https://ssrn.com/abstract=4842761 or http://dx.doi.org/10.2139/ssrn.4842761*

42. *Acharya, Kamal, Web Chatting Application Management System Project Report (April 25, 2022). Available at SSRN: https://ssrn.com/abstract=4842771 or http://dx.doi.org/10.2139/ssrn.4842771*

43. Acharya, Kamal, Automobile management system project report (May 25, 2022). Available at SSRN: https://ssrn.com/abstract=4846917 or http://dx.doi.org/10.2139/ssrn.4846917

44. *Acharya, Kamal, College bus management system project report (April 25, 2023). Available at SSRN: https://ssrn.com/abstract=4846920 or http://dx.doi.org/10.2139/ssrn.4846920*

45. *Acharya, Kamal, Courier management system project report (May 25, 2023). Available at SSRN: https://ssrn.com/abstract=4846922 or http://dx.doi.org/10.2139/ssrn.4846922*

46. *Acharya, Kamal, Event management system project report (April 25, 2021). Available at SSRN: https://ssrn.com/abstract=4846927 or http://dx.doi.org/10.2139/ssrn.4846927*

47. Acharya, Kamal, Library management system project report II (May 25, 2020). Available at SSRN: https://ssrn.com/abstract=4848857 or http://dx.doi.org/10.2139/ssrn.4848857

48. Kamal Acharya. Online directory management system project. Authorea. Jun 11, 2025 DOI: https://doi.org/10.22541/au.174965566.61058065/v1